Time Series Retrieval using DTW-Preserving Shapelets

Ricardo Carlini Sperandio[1], Simon Malinowski[2], Laurent Amsaleg[3], and Romain Tavenard[4]

[1] IRISA-Inria, France
[2] IRISA-Univ. Rennes 1, France
[3] CNRS-IRISA, France
[4] Univ. Rennes 2, France

Abstract. Dynamic Time Warping (DTW) is a very popular similarity measure used for time series classification, retrieval or clustering. DTW is, however, a costly measure, and its application on numerous and/or very long time series is difficult in practice. This paper proposes a new approach for time series retrieval: time series are embedded into another space where the search procedure is less computationally demanding, while still accurate. This approach is based on transforming time series into high-dimensional vectors using DTW-preserving shapelets. That transform is such that the relative distance between the vectors in the Euclidean transformed space well reflects the corresponding DTW measurements in the original space. We also propose strategies for selecting a subset of shapelets in the transformed space, resulting in a trade-off between the complexity of the transformation and the accuracy of the retrieval. Experimental results using the well known UCR time series demonstrate the importance of this trade-off.

1 Introduction

Time series data is massively produced 24*7 by millions of users, worldwide, in domains such as finance, agronomy, health, earth monitoring, weather forecasting, multimedia, etc. Due to the advances in sensor technology and its proliferation, applications may produce millions to trillions of time series per day, making time series data-mining further challenging.

Most time series data-mining algorithms rely on the Dynamic Time Warping (DTW) [16] measure at their core, which proves to return high-quality results, but which is also very costly to compute [3]. Many researchers have attempted to reduce its cost in order to run it at scale. Numerous mart optimizations, diverse lower bounds and other techniques have been applied [4], however, the quest for processing extremely large collections of time series is still active.

This paper focuses on the task of time series retrieval according to the DTW measure. The classical scenario for this task is the following: let τ be a test time series (query), retrieval is about finding a time series in a dataset $\mathcal{T}$ which is the

closest one to τ with respect to the DTW measure. In other words, it is finding T^*, such that:

$$T^* = \arg\min_{T_i \in \mathcal{T}} DTW(\tau, T_i) \quad (1)$$

A traditional way to identify T^* relies on the brute force DTW computation between τ and all series of $\mathcal{T}$. This approach is not tractable when dealing with long time series or huge data sets. Hence, approximated methods are preferred, aiming at reducing the retrieval costs while being as accurate as possible.

We propose here an approximate time series retrieval approach based on the shapelet transform. The basic idea of the proposed approach is to transform the time series of the dataset into vectorial representations such that a Euclidean search can then be efficiently applied to retrieve the nearest neighbour of the transformed query. Of course, the transformation needs to be carefully designed so that the approximate search is accurate enough. Another crucial point is related to the computing cost of the transformation. At test time, the query needs to be first transformed before being compared with transformed time series of the dataset. Hence, the transformation should not be too costly.

In this paper, DTW-preserving shapelets [11] are used to transform time series to a vectorial representation. This transform is such that the relative Euclidean distance in the transformed space well reflects the original DTW measurements. They are hence well adapted to our task. Transforming time series has a cost, but it can be traded-off against the accuracy of the retrieval by selecting a varying number of shapelets for computing the vectorial representations. Three shapelet selection strategies are proposed in this paper.

In addition, we propose a shapelet selection procedure in order to identify and rank shapelets that are more suited to the retrieval task (via a dedicated measure). This step has two purposes: reducing the number of shapelet for the transformation of the query (to save computing time), while keeping a good trade-off with accuracy of the retrieval.

The remainder of this paper is organized as follows: Section 2 presents related work. The proposed Learning DTW-Preserving Shapelet Retrieval (DPSR) method is presented in Section 3. Section 4 presents experimental results and conclusions and future work are presented in Section 5.

2 Related work

Various similarity measures have been used for time series retrieval, and excellent surveys provide a good coverage of their pros and cons [4, 20]. In a nutshell, the straightforward Euclidean distance is not robust to distortions and time shifts, and is hence not very appropriate for time series. Approaches based on the DTW are much preferred because it is able to find the optimal alignment between two given time series, thus coping with local distortions along the time-line. DTW, however, is costly to calculate because of its quadratic time complexity.

Several approaches have therefore been proposed in order to speed up the computation of DTW, including restricting to the diagonal of the distance

matrix [7, 16] or the design of lower bounds. The most popular lower bound is the LB_Keogh lower bound (LB_Keogh) [8]. It first needs to rescale each time series to the same length and then it builds their envelopes by accounting for their maximum and minimum values inside a sliding window. The distance between two time series returned by LB_Keogh corresponds to the areas of their envelopes that do not overlap. In [14], Rakthanmanon *et al.* proposed the UCR Suite framework which includes several acceleration approaches that can be combined in order to index time series under the DTW measure. Recently, Tan *et al.* in [18] proposed an adaptation of Priority Search K-means to index time series embedded in a space induced by DTW. Interested readers should read three comprehensive reviews [3, 4, 13].

The Piecewise Aggregate Approximation (PAA) [9, 22] has been used to build iSAX [17], one of the most famous time series indexing system. PAA is a transformation that divides time series into smaller pieces and create an approximation of each piece. This transformation is at the core of iSAX, which was shown to be very accurate and fast for retrieving time series. However, it is based on the euclidean distance and not the DTW.

We propose a retrieval scheme based on a transformation preserving the DTW. This transformation makes use of shapelets. Shapelets were introduced by Ye and Keogh in [21] for time series classification. The underlying intuition behind shapelets is that time series belonging to one class are likely to share some common subsequences. Shapelets were therefore originally defined as existing subsequences of time series that best discriminate classes. Hills *et al.* then build on the idea of shapelets by proposing the Shapelet Transform [6]. In this approach, each time series is transformed into a vector whose components represent the distances between the time series and the shapelets. Transforming a time series into its vectorial representation requires to slide each shapelet against that time series in order to find the best matching locations and then compute the corresponding distances. The cost of shapelet transform is therefore highly dependant on the number of shapelets that are used to create vectorial representations. Instead of using existing subsequences as shapelets, Grabocka *et al.* in [5] propose to rather forge the shapelets by learning the subsequences that minimize a classification loss. Shapelets have also been used for unsupervised tasks and not only for classification. In [11], Lods *et al.* learn shapelets such that the resulting vectorial representation preserves as well as possible the DTW distance between raw time series, targeting time series clustering. In [23], Zakaria *et al.* extract the shapelets dividing the set of time series into well separated groups.

3 Time series retrieval with DTW-preserving shapelets

In this section, we detail our approach for time series retrieval under DTW. This approach builds on (i) the shapelet transform paradigm and (ii) the DTW-preserving shapelets that are proposed in [11]. We first quickly review these building blocks before presenting the design of our retrieval scheme.

3.1 Background on shapelets and shapelet transform

A shapelet $S = s_1, \ldots, s_l$ is a temporal sequence (that can be extracted from existing time series, or learned). The distance between S and a time series $T = t_1, \ldots, t_L$ is defined as:

$$d(T, S) = \min_{1 \leq j \leq L-l+1} \sqrt{\sum_{i=1}^{l} (s_i - t_{i+j-1})^2} \tag{2}$$

In other words, Euclidean distances between S and every subsequence of T (of length l) are computed and only the best match (minimum distance) is kept. The shapelet transform of a time series was proposed in [6] for time series classification. It is a two step process: (i) selecting an appropriate set of shapelets $\mathcal{S} = \{S_1, \ldots, S_K\}$ and (ii) transforming time series into Euclidean vectors. During the second step, each time series T is transformed into a vector $v_1, \ldots, v_K$ such that $v_i = d(T, S_i)$, $1 \leq i \leq K$, where $S_1, \ldots, S_K$ are the shapelets that were selected during the first step. The dimensions of v represent the distance between T and the shapelets of $\mathcal{S}$. Such representations of time series are then used to feed a classifier, when the targeted application is classification.

3.2 Learning DTW-preserving shapelets

In [11], Lods *et al.* propose to learn a set of shapelets such that the shapelet-transform representation preserves as well as possible the original DTW measure. The shapelets are learned such that Euclidean distance in the transformed space approximates the DTW. Such shapelets can hence be used for unsupervised tasks, that is, where no labels are available.

The approach proposed by Lods *et al.* relies on minimizing the loss:

$$\mathcal{L}(T_i, T_j) = \frac{1}{2} \left(\mathrm{DTW}(T_i, T_j) - \beta \left\| \overline{T}_i - \overline{T}_j \right\|_2 \right)^2, \tag{3}$$

where β is a scale parameter, learned. The overall loss for a dataset $\mathcal{T}$ of N time series is given by:

$$\mathcal{L}(\mathcal{T}) = \frac{2}{N(N-1)} \sum_{i=1}^{N} \sum_{j=i+1}^{N-1} \mathcal{L}(T_i, T_j). \tag{4}$$

The minimization of this loss is done via a stochastic gradient descent with respect to β and $\mathcal{S}$. Once the shapelets and the parameter β are learned, they can be used to transform every time series in a Euclidean vector.

3.3 Transforming times series for retrieval under DTW

In this paper, we propose a retrieval scheme based on transforming time series into high-dimensional vectors. Its offline step transforms all time series of the data

set into their vectorial representation, as explained above. Its online step uses a query to probe the dataset. To do so, the vectorial representation of the query is first determined. Then, Euclidean distances to the transformed times series of the dataset are computed. This results in a list of transformed time series that are ranked according to their proximity to the transformed query. The nature of the transformation that is using DTW-preserving shapelets is such that this ranking in the transformed space is an approximation of the ranking that would be produced in the original space according to the DTW measure. However, this approximate ranking is obtained much faster, as Euclidean measurements are cheaper to obtain compared to DTW measurements.

Two ways to process that ranked list of time series in the transformed space can be designed: (a) the original time series associated to the top-ranked in this list is considered to be the nearest neighbour of the query, or (b) the true DTW is computed between the original untransformed query and the original version of the first few elements of that list in order to refine the search. (a) puts a lot of pressure on the quality of the shapelet transform because the nearest time series under DTW has also to be the closest in the Euclidean space, and this for any time series in the dataset and any query. Odds of degrading quality in comparison to what the true DTW would determine are high, but this method is extremely fast. In contrast, (b) is more demanding because more DTW are computed, but it is also returns better quality results as more time series are scrutinized. In this case, it matters that the closest time series under DTW belongs to these first few elements, instead of being ranked first.

Overall, two properties are important for an approximate retrieval scheme: (i) the complexity of the transformation should be small to reduce the overhead induced by transforming the query and (ii) the true nearest neighbour has to be as close as possible to the first element of the list of approximate nearest neighbours. In other words, the transformation should preserve the ranking.

The retrieval scheme proposed in this paper focuses on these two important properties. The use of DTW-preserving shapelets ensures that the Euclidean search in the transformed space mimics the search in the original space.

3.4 Ranking and selecting shapelets

The transformation of the query before searching in the transformed space is costly. Indeed, the computation of one coordinate of the transformed vector requires sliding a shapelet over the query τ and finding the best match. It is a costly operation for long time series and when the number of learned shapelets is high. We hence focus in this section on selecting and ranking a subset of shapelets most suited to the retrieval task.

General considerations about high-dimensional representations suggest that components of vectors might not all be equally useful. This well known observation led to designing dimensionality reduction methods performing feature selection. Feature selection algorithms typically use (i) an evaluation metric to compare different feature subsets, (ii) a strategy for building consecutive subsets and (iii) a stopping criterion [1,10,12,15]. The characteristics that we use to design our

shapelet selection algorithm are described hereafter. Selecting a few appropriate shapelets saves computations at transform time without degrading significantly the quality of the approximation. Algorithm 1 is the corresponding pseudo-code.

Evaluation metric to compare shapelet subsets To compare the performance of different shapelet subsets, we need a groundtruth based on the true DTW between time series. To build that groundtruth, the DTW between all time series pairs in the training set is computed and we record for each time series the identifier of its nearest-neighbour. This is done once only, off-line.

We use a 10-fold validation setup in order to evaluate the performance of a subset. For one transformed time series in the validation set (query), we rank all the transformed time series in the training set according to their Euclidean distance to the query. It is therefore possible to determine at which rank the true 1-nearest neighbor time series is. Repeating this operation for all the validation time series and for all the folds amounts to building an histogram of the rank at which the true nearest-neighbour appear. This histogram can also be interpreted as the empirical probability of observing the true 1-nearest neighbor time series at any particular rank after the transform. This histogram can therefore be considered to be a probability density function (after a proper normalization though). From this PDF, it is straightforward to construct its natural counterpart which is the CDF, the cumulative distribution function, and to compute the associated area under the curve (AUC). We consider this AUC value as the performance measure to evaluate the quality of a shapelet subset. The higher that AUC, the better the shapelet subset. This metric is well adapted to the task of nearest-neighbour retrieval as it favors high ranking of the true nearest neighbor in the approximated list.

Shapelet subset selection To select the best subset of shapelets, an exhaustive selection method can be applied. However, in this case, the computational cost is prohibitively high. We have chosen a greedy-based forward selection method, that is classically used in the feature selection domain.

Our procedure to select the best shapelet subset begins with an empty list. Then it iteratively adds shapelets that best improve the quality of the subset (by measuring the resulting AUC), one by one, until a stopping criterion is met (the different stopping criterion we used are described in the following).

Stopping criterion We define three different stopping criterion, that determine when to stop adding shapelets to the current set of selected shapelets:

1. Global maximum ($DPSR_g$): Shapelets are added one by one until no more shapelets are available. At the end, the subset that leads to the best overall AUC is selected.
2. $tan(x) < 1$ ($DPSR_t$): We compute the normalized slope between the AUC value of the current selected subset and the one obtained by adding the shapelet that best improves the AUC. If this slope is less than 1, then the shapelet selection is stopped.

3. Local maximum ($DPSR_l$): The shapelet selection is stopped as soon as adding a shapelet does not improve the AUC value.

Algorithm 1 Shapelet ranking and selection

input: $\mathcal{S}$ {Set of learned shapelets}, $\mathcal{T}$ {Time series train set}
output: $\mathcal{S}_s$ {Ranked list of selected shapelets}
$\mathcal{S}_a \leftarrow \mathcal{S}$ {Shapelets available to evaluate, initially all}
$\mathcal{S}_s \leftarrow \emptyset$ {Shapelets selected, initially empty}
$stop \leftarrow$ `FALSE`
repeat
  $score_b \leftarrow -1$
  for all $S \in \mathcal{S}_a$ **do**
    $\mathcal{S}_t \leftarrow \mathcal{S}_s \cup S$, $score \leftarrow$ AUC value of the set $\mathcal{S}_t$
    if $score > score_b$ **then** $score_b \leftarrow score$, $S_b \leftarrow S$
  end for
  $\mathcal{S}_s \leftarrow \mathcal{S}_s \cup S_b$, $\mathcal{S}_a \leftarrow \mathcal{S}_a \setminus S_b$
  if Stopping criterion is met **then** $stop \leftarrow$ `TRUE`
until $stop =$ `TRUE`

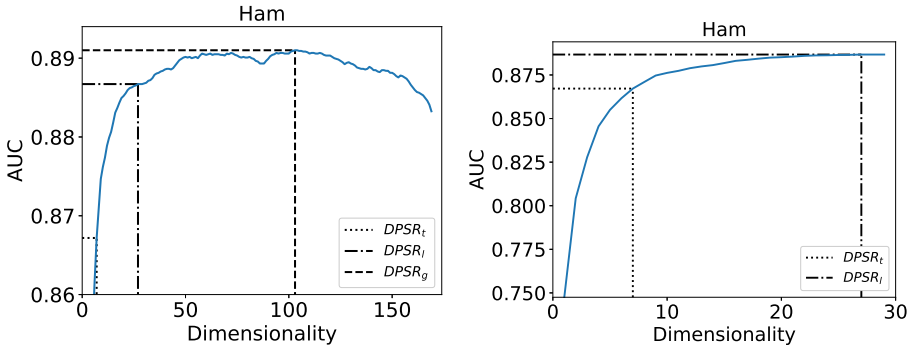
Figure 1 shows the AUC values at each iteration of the shapelet selection algorithm on the `Ham` dataset (from the UCR-UEA archive [2]). For this dataset, it has been learned 170 shapelets using Lods *et al.* approach ($DPSR_f$). We can see on this figure the impact of the 3 different criteria. If the tangent criterion ($DPSR_t$) is used, then the selection process would end with 7 shapelets (Fig. 1b, which zooms on the first 30 dimensions), while 27 would be selected by the local maximum criterion ($DPSR_l$) and 103 by the global maximum one ($DPSR_g$). Note that for the global maximum criterion the selection process cannot be stopped before having ranked and selected all shapelets one by one, contrary to the two other criteria for which the process can be stopped as soon as the criterion is reached. We illustrate in the experiments the trade-off between dimensionality and accuracy induced by these criteria.

4 Experiments

This section presents an experimental evaluation of our DPSR approach. The trade-off between accuracy and computation complexity is discussed. Performance of DPSR is compared to Piecewise Aggregate Approximation (PAA) [9, 22] and LB_Keogh lower bound (LB_Keogh) [8]. We start with our experimental setup.

4.1 Experimental setup

We consider the 85 datasets from the well known UCR-UEA Time Series Archive [2]. For all these datasets, we use test sets series as queries.



(a) (b)

Fig. 1: (a) Impact of the three different stopping criterion in terms of number of selected shapelets for the Ham dataset. (b) Zoom on the first dimensions of (a).

Before running any experiment, we built a full DTW-based groundtruth. The true DTW measurements between all time series pairs in each of the 85 families are determined. From these measurements, it is straightforward to derive the nearest-neighbour of each time series.

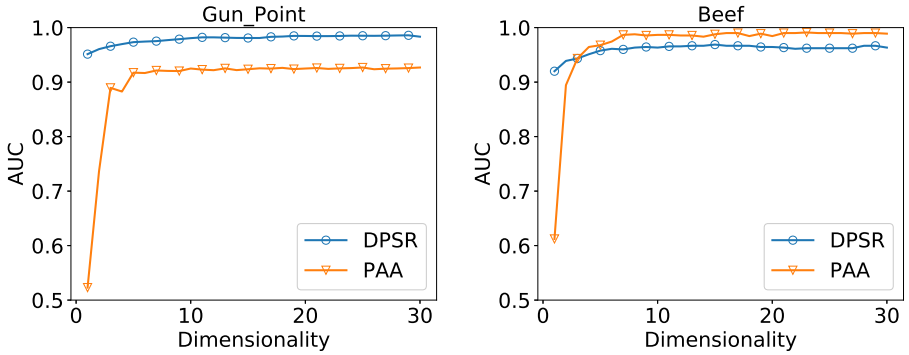
The feature selection algorithm for DTW-preserving Shapelets that is presented in this paper is evaluated against two solid competitors that are (i) the LB_Keogh lower-bound used to accelerate the computation of the true DTW (used in the UCR Suite framework) and (ii) the PAA approach (on which iSAX is based).

Our experiments are performed on a 24-core 2.8 GHz Intel Xeon ES-2630 with 64 GB of memory. All algorithms and structures are implemented in Python3, Cython and NumPy. Although the machine has 24 cores, the only operations using parallelism are the distance computations handled by NumPy during the course of each experiment – no other parallelism is enforced. The tslearn [19] toolkit was used for the computation of PAA and LB_Keogh.

For each family in the UCR archive, a set of DTW-preserving shapelets is learned using the algorithm proposed in [11], with default parameters. To learn high-quality shapelets, 500,000 iterations of the gradient descent algorithm are performed. In addition, two sets of shapelets are learned for each family of time series, and the one with the smallest overall loss is selected.

4.2 Dimensionality versus accuracy: reaching a plateau

This first experiment aims at comparing the performance of PAA and DPSR for time series retrieval. This comparison is performed in terms of trade-off between AUC and the dimensionality of the transformation. Transformations range from very rough approximations (few shapelets, few pieces for PAA) to finer grain representations. For consistency, we compared PAA and DPSR for the same



(a) Gun Point (b) Beef

Fig. 2: Comparing DPSR and PAA for two specific time series. The ability to retrieve the correct time series from the groundtruth is represented by the resulting AUC.

Dimensionality	5	10	20	30
# DPSR wins	70	69	73	72
# PAA wins	15	16	12	13
Average AUC for PAA	0.838	0.847	0.844	0.841
Average AUC for DPSR	0.906	0.921	0.929	0.932

Table 1: Comparing DPSR and PAA for the full UCR Archive at different dimensionalities. Number of times each method outperforms the other in terms of AUC is reported together with the average AUC values over the 85 datasets.

dimensionality. Figure 2 illustrates the resulting AUC values for two specific data sets, `Gun_Point` and `Beef`. With `Gun_Point`, DPSR outperforms PAA for all dimensionalities. The results for `Beef` are more contrasted: when the time series is split into more than 5 pieces, PAA outperforms the shapelet based approach. Please note that these figures show quality measures for the first 30 dimensions only. Considering more than 30 dimensions does not provide any useful extra information for these datasets.

Table 1 compiles the comparison of PAA and the approach based on DPSR for all 85 time series by counting the number of times each method performs better than the other, for dimensionalities 5, 10, 20 and 30. We also report the average AUC values at these dimensionalities for the two different approaches. Overall, this table shows that DPSR consistently outperforms PAA for all dimensionalities. We can also observe that both methods seem to reach an AUC plateau, sooner for PAA than it is for DPSR.

For DPSR, this plateau highlights the importance of selecting shapelets. It indicates that the original approach by Lods *et al.* creates far too many shapelets,

	DPSR_t	DPSR_l	DPSR_g	DPSR_f
Avg. AUC	0.906	0.928	0.935	0.934
Avg. Dim.	4.4	27.3	54.0	156.1

Table 2: Average AUC and dimensionalities for feature subset selection strategies.

and that a lot of them do not contribute significantly to enhancing the resulting vectorial representation. Generating so many shapelets is wasting computing resources at transform time because many shapelets have to be slid, some in pure waste. This is particularly important at query time, as reducing the cost of transforming the query time series is paramount.

4.3 Shapelet selection strategies for DPSR

In Section 3.4, three strategies for stopping aggregating selected features were presented. We now evaluate their effectiveness, which is a trade-off between their accuracy in terms of AUC and the transformation cost they cause. Typically, small subsets allow for very fast transforms (just a few shapelets need to be slid at test time) but quality is typically low, whereas in contrast larger subsets improves AUC performance but cause more expensive transform operations.

To observe this trade-off, we selected by cross-validation on the training sets of each dataset the best shapelet subset for the three different stopping criterion. We then used these subsets at test time for approximate retrieval. The AUC performance of the three stopping strategies has been computed on the 85 considered datasets. The average AUC value is given in Table 2, together with the average number of selected shapelets. The last column of Table 2 (DPSR_f) corresponds to the case where no shapelet selection is performed (i. e., the whole shapelet set learned beforehand is used).

We can observe that the three proposed criteria generate a trade off between accuracy of the retrieval (AUC) and computational time of the transform (linear with the number of shapelets). DPSR_t, the most aggressive strategy, selects very few shapelets (a little bit more than 4, on average) for an average AUC of 0.906. DPSR_g, the most conservative strategy uses on average 54 shapelets, but the corresponding quality improvement is quite small: it goes from 0.906 to 0.935. This is a clear illustration of the trade off, also exemplified by the DPSR_l strategy, which is in between these two strategies.

We can also observe the importance of the feature selection itself: when no selection is made (DPSR_f), the average AUC is slightly lower than for DPSR_g and the corresponding average number of shapelets used is almost tripled.

4.4 Feature selection versus constrained feature learning

The previous experiment demonstrated that only a small fraction of the learned shapelets are truly useful because they significantly contribute to the quality of

	DPSR vs. PAA		DPSR vs. LB_Keogh	
	# wins for DPSR	# wins for PAA	# wins for DPSR	# wins for LB_Keogh
$DPSR_t$	54	31	33	52
$DPSR_l$	63	22	62	23
$DPSR_g$	70	15	64	21

Table 3: Comparing DPSR, PAA and LB_Keogh with their best parameters (learned by cross-validation). We report here the number of times each method outperforms the other in terms of AUC.

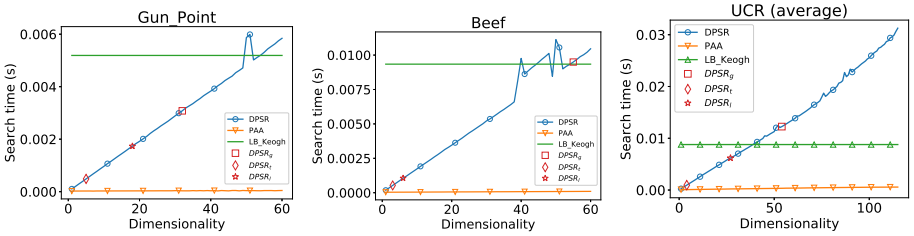
the retrieval. We show here that it is the combination of the learning stage and the feature selection strategy that leads to such behaviour. For that purpose, we compare the AUC performance of the proposed retrieval scheme ($DPSR_t$ and $DPSR_l$) with a method where the same number of shapelets is directly learned using the algorithm presented in [11].

We used the previous experiment to record for each dataset how many shapelets the $DPSR_t$ and $DPSR_l$ selected. Then, we ran the shapelet learning algorithm of [11] using that number of shapelets (for each dataset and for each DPSR strategy). We know from the previous experiments that the average AUC for $DPSR_t$ over all the datasets is 0.906, with using 4.4 shapelets on average. When directly learning that same number of shapelets, then the average AUC over all datasets is 0.866. Furthermore, considering individually the 85 families of time series, the $DPSR_t$ strategy performs better than the one directly learning the appropriate number of shapelets in 74 cases (out of 85). Same conclusions can be drawn with $DPSR_l$. The average AUC of $DPSR_l$ is 0.928, whereas the AUC obtained when directly learning the same number of shapelets is equal to 0.905. The $DPSR_l$ strategy wins 71 times out of 85 in that case.

These results indicate that it is worth spending more time offline to learn a huge set of shapelets and then selecting the more appropriate. This is better than trying to save computational time by learning less shapelets. Also, our feature selection strategy allows to decide on the number of shapelets in a data driven fashion, contrary to a purely heuristic approach.

4.5 Comparing methods at their best

In this experiment, we compare the respective performance of DPSR, PAA and LB_Keogh when their parameters (number of segments for PAA, subset of shapelets for DPSR and window length for LB_Keogh) are cross-validated on the train set. The results comparing the performance of (i) DPSR and PAA methods and (ii) DPSR and LB_Keogh are given in the Table 3. This table gives the number of times each method wins over the other. Overall, DPSR outperforms PAA even for the $DPSR_t$ criterion. Interestingly, comparing the dimensionalities when DPSR or PAA are winning provides insightful results. Consider for example the $DPSR_t$ strategy. That strategy wins over PAA 54 times. Among these 54



(a) Gun_Point (b) Beef (c) UCR

Fig. 3: Average Search Times for two specific time series and for the full UCR Archive. Varying dimensionality.

wins, in 45 cases, DPSR needs fewer dimensions than PAA. It means that in 9 cases, DPSR needs more dimensions than PAA to provide better results. The dual point of view is also insightful: PAA outperforms the DPSR_t method in 31 cases, but all PAA transformations need more pieces than DPSR_t. Not only DPSR wins more frequently than PAA, but when it wins, it is with shorter representations. This is also true for DPSR_l and DPSR_g. The average AUC value for PAA is equal to 0.866 which is worse than the average AUC value of the 3 DPSR approaches. Against LB_Keogh, DPSR is only worse for the DPSR_t criterion which leads to a very small representation. The average value of AUC for LB_Keogh is 0.908. It is better than DPSR_t, but not than DPSR_l and DPSR_g.

An important observation is that, unlike LB_Keogh and PAA, DPSR allows comparison between time series of different lengths.

4.6 Search costs

So far, only quality comparisons have been done. We observe now the computational costs of the approaches discussed here. Additional experiments were performed for recording search times when the dimensionality of the representations for DPSR and PAA gradually increase. Search times for `Gun_Point` and `Beef` are plot on Figure 3 (the plots show only results for their first 60 dimensions), and the average search times for the full UCR archive is on Figure 3c.

These figures also show the time it takes to compute only the envelopes of time series for LB_Keogh. That process, coupling LB_Keogh and DTW is guaranteed to find the same time series as the one indicated in the groundtruth. The quality of any approximate search scheme can only be equal or lower. But the time for solely computing the LB_Keogh value on all time series is the absolute minimal cost the real LB_Keogh+DTW could have.

These figures show that the time for computing envelopes with LB_Keogh is fixed, which is normal. They also show that PAA is the fastest approach. Its underlying principles are simple and cause light computations, the search times increasing slightly with the dimensionality. The time taken per search for DPSR is

also increasing with the number of dimensions: there are more and more shapelets to slide, and distance computations are more demanding. Three remarkable signs are placed on the search time plot for the DPSR approach. They refer to the search times observed when the number of shapelets in use correspond to what DPSR_t, DPSR_l and DPSR_g determined.

5 Conclusions and future work

In this work, we have presented an approach for time series retrieval based on learning DTW-preserving shapelets (DPSR). This approach first transforms time series into high dimensional vectors such that the Euclidean distance between the vectors in the transformed space well reflects the DTW measurements in the original space. This targets preserving the quality of time series retrieval compared to the DTW. Relying on Euclidean distances is more efficient than computing the costly DTW measures. This targets computational efficiency, facilitating time series retrieval at scale.

In order to cope with with larger scales, we also propose different shapelet selection strategies to trade complexity of the retrieval against accuracy. Even the most aggressive strategy (that select very few shapelets) provides reasonable accuracy. Experimental results show the importance of this feature selection.

This work is a first step into the design of a time series indexing system. At very large scale, the many high dimensional vectors representing time series could be inserted into an index, avoiding the exhaustive Euclidean distance calculations, further improving performance. Such an approach can be advantageously used for *anytime indexing* of time series.

Acknowledgments The current work has been performed with the support of CNPq (*Conselho Nacional de Desenvolvimento Científico e Tecnológico*), Brazil (Process number 233209/2014-0). The authors are grateful to the TRANSFORM project funded by STIC-AMSUD (18-STIC-09) for the partial financial support to this work.

References

1. Blum, A., Langley, P.: Selection of relevant features and examples in machine learning. Artif. Intell. 97(1-2), 245–271 (1997)
2. Chen, Y., Keogh, E., Hu, B., Begum, N., Bagnall, A., Mueen, A., Batista, G.: The UCR time series classification archive (July 2015), `www.cs.ucr.edu/~eamonn/time_series_data/`
3. Ding, H., Trajcevski, G., Scheuermann, P., Wang, X., Keogh, E.J.: Querying and mining of time series data: experimental comparison of representations and distance measures. PVLDB 1(2), 1542–1552 (2008)
4. Esling, P., Agón, C.: Time-series data mining. CSUR 45(1), 12:1–12:34 (2012)
5. Grabocka, J., Schilling, N., Wistuba, M., Schmidt-Thieme, L.: Learning time-series shapelets. In: KDD. pp. 392–401. ACM (2014)

6. Hills, J., Lines, J., Baranauskas, E., Mapp, J., Bagnall, A.: Classification of time series by shapelet transformation. DMKD 28(4), 851–881 (2014)
7. Itakura, F.: Minimum prediction residual principle applied to speech recognition. IEEE Trans. Signal Processing 23(1), 67–72 (1975)
8. Keogh, E.J.: Exact indexing of dynamic time warping. In: VLDB. pp. 406–417. Morgan Kaufmann (2002)
9. Keogh, E.J., Chakrabarti, K., Pazzani, M.J., Mehrotra, S.: Dimensionality reduction for fast similarity search in large time series databases. KAIS 3(3), 263–286 (2001)
10. Li, J., Cheng, K., Wang, S., Morstatter, F., Trevino, R.P., Tang, J., Liu, H.: Feature selection: A data perspective. ACM Comput. Surv. 50(6), 94:1–94:45 (2017)
11. Lods, A., Malinowski, S., Tavenard, R., Amsaleg, L.: Learning dtw-preserving shapelets. In: IDA. LNCS, vol. 10584, pp. 198–209. Springer (2017)
12. Moradi, P., Rostami, M.: A graph theoretic approach for unsupervised feature selection. Eng. Appl. of AI 44, 33–45 (2015)
13. Papapetrou, P., Athitsos, V., Potamias, M., Kollios, G., Gunopulos, D.: Embedding-based subsequence matching in time-series databases. TODS 36(3), 17:1–17:39 (2011)
14. Rakthanmanon, T., Campana, B.J.L., Mueen, A., Batista, G., Westover, M.B., Zhu, Q., Zakaria, J., Keogh, E.J.: Searching and mining trillions of time series subsequences under DTW. In: KDD. pp. 262–270. ACM (2012)
15. Saeys, Y., Inza, I., Larrañaga, P.: A review of feature selection techniques in bioinformatics. Bioinformatics 23(19), 2507–2517 (2007)
16. Sakoe, H., Chiba, S.: Dynamic programming algorithm optimization for spoken word recognition. IEEE Trans. Signal Processing 26(1), 43–49 (1978)
17. Shieh, J., Keogh, E.J.: *i*sax: indexing and mining terabyte sized time series. In: KDD. pp. 623–631. ACM (2008)
18. Tan, C.W., Webb, G.I., Petitjean, F.: Indexing and classifying gigabytes of time series under time warping. In: SDM. pp. 282–290. SIAM (2017)
19. Tavenard, R.: tslearn: A machine learning toolkit dedicated to time-series data (2017), `https://github.com/rtavenar/tslearn`
20. Wang, X., Mueen, A., Ding, H., Trajcevski, G., Scheuermann, P., Keogh, E.J.: Experimental comparison of representation methods and distance measures for time series data. DMKD 26(2), 275–309 (2013)
21. Ye, L., Keogh, E.J.: Time series shapelets: a new primitive for data mining. In: KDD. pp. 947–956. ACM (2009)
22. Yi, B., Faloutsos, C.: Fast time sequence indexing for arbitrary lp norms. In: VLDB. pp. 385–394. Morgan Kaufmann (2000)
23. Zakaria, J., Mueen, A., Keogh, E.J.: Clustering time series using unsupervised-shapelets. In: ICDM. pp. 785–794. IEEE C. Society (2012)